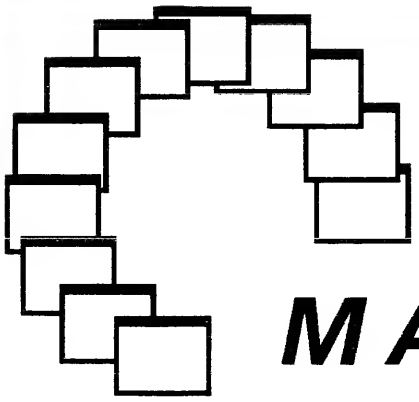
Masterscope (c) Xerox 1985



Volume 1, Number 3

August 1985

MASTERSCOPE

EDITORS' MESSAGE

Xerox has recently announced a major reallignment of its Xerox Systems Group. Under XSG, a new organization called the Information Systems Division was formed. The AI Systems Business Unit, formerly reporting to Xerox Special Information Systems, has been renamed to Xerox Artificial Intelligence Systems and reports directly to the Information Systems Division. We view this reorganization as very positive both to you, our customers, and to our business outlook in general.

With this issue, we are introducing a new column: "Programmer's Corner: Program and System Tips." It is intended as feedback from the Software Support group to you, briefly explaining some of the more frequently asked questions.

As a reminder, we are having our first International Users' Group meeting at IJCAI, starting at 5:00 p.m. on Wednesday, August 21st. We hope that you will all be able to attend. Space is limited, so sign up soon. The sign-up sheet for the meeting is included at the end of this issue of Masterscope.

We are looking forward to seeing you at IJCAI. Please come by our booth and hospitality suite for some exciting new announcements.

The Editors

Masterscope is edited and formatted on a Xerox 1108 using TEdit and Sketch, with camera-ready copy prepared using a Xerox 8045.

Interlisp-D

SpinPro ™ : an Expert System for Optimizing Ultracentrifuge Runs

by Matt Heffron and Phil Martz
Beckman Instruments, Inc.

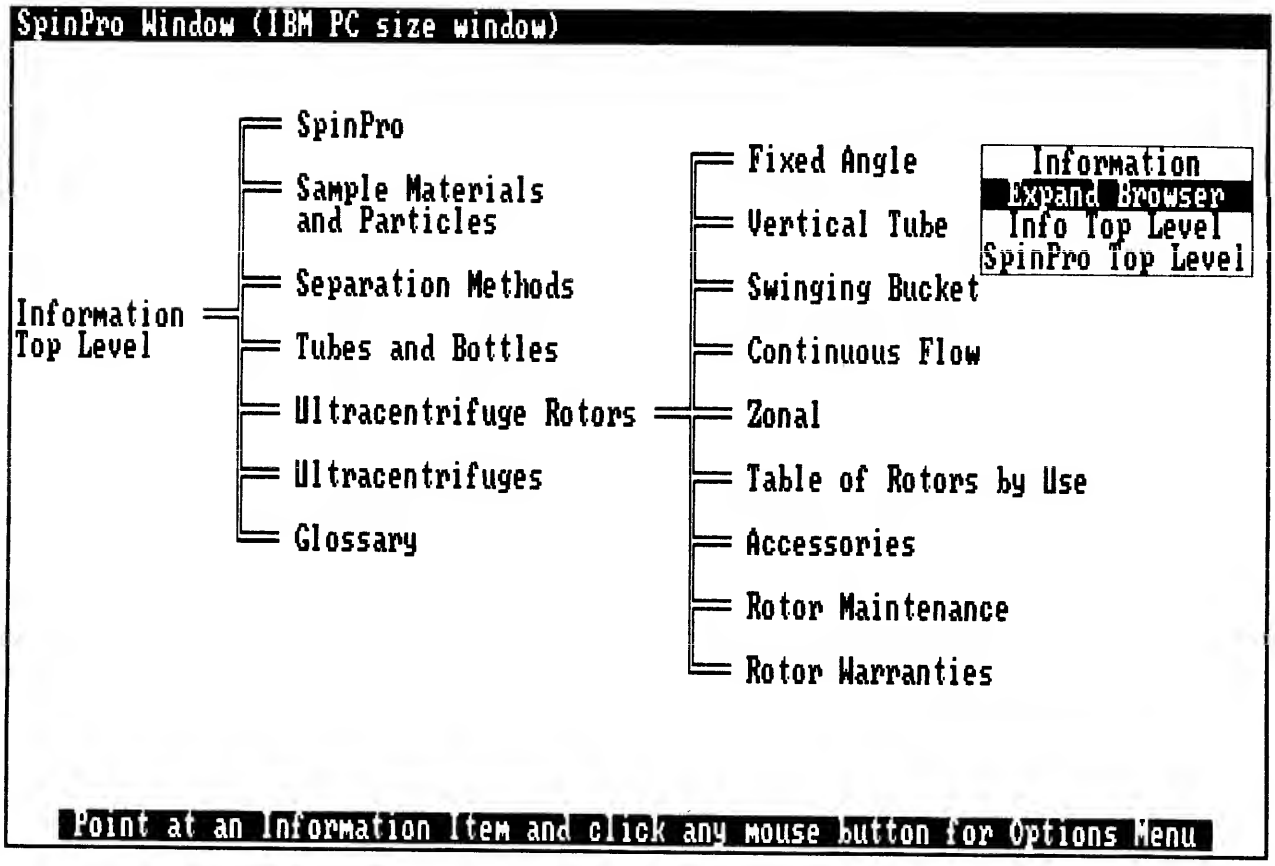
Ultracentrifugation is a common method in the separation of biological materials. Despite its widespread use, few investigators fully exploit the capabilities of ultracentrifugation. As a result, run times may be unnecessarily long, separations may be indistinct, or the run itself may be ineffective. Time and materials are wasted. Beckman Instruments has long been a pioneer and leading manufacturer of ultracentrifuge equipment, and Beckman's scientists have acquired a lot of expert knowledge in their application. These factors led to Beckman's development of the SpinPro Ultracentrifugation Expert System.

SpinPro is an expert system which designs ultracentrifugation procedures. The investigator and the program participate in a question and answer dialogue in which the research goals and sample characteristics are defined. This interaction with the program is modeled on the way customers currently interact with Beckman's human experts by telephone. At the conclusion of the dialogue, SpinPro produces four reports. One report summarizes the questions posed by SpinPro and the answers provided by the investigator. The second report describes an optimal ultracentrifugation procedure to achieve the stated goals using the optimal equipment. The third report is similar, but outlines a procedure based upon, and constrained by, the centrifuge equipment available in the investigator's laboratory. A fourth report compares the two procedures and the effectiveness of each in performing the run. Thus, the program performs the role of an expert advisor, offering knowledgeable advice and comparing alternatives.

The SpinPro Ultracentrifugation Expert System was developed in Interlisp (Interlisp-D on a Xerox 1108, and Interlisp-10 on a DECSYSTEM-2060), and runs under Common Lisp on the IBM-PC/XT. The inferencing engine for SpinPro was developed in-house since we needed to have flexibility in the choice of target systems, and control over the porting process. Since the targeted users of this system are research biologists, they are not necessarily inclined to want to learn to use a computer system that is complicated. A great deal of effort was spent in developing a good user interface. Interlisp-D provided the inspiration and modeling environment for the mouse-based interface that we chose. We modeled the IBM-PC interface on the Xerox 1108 by building a window which was the same size as the IBM-PC (25 rows x 80 characters). We also created a font by selecting characters from the IBM-PC's ROM character generator. We then wrote a simple form of the Lisp Library GRAPHER package which we could port to the IBM-PC. Selection of SpinPro's major functions is made with the mouse from a simple graph form of menu.

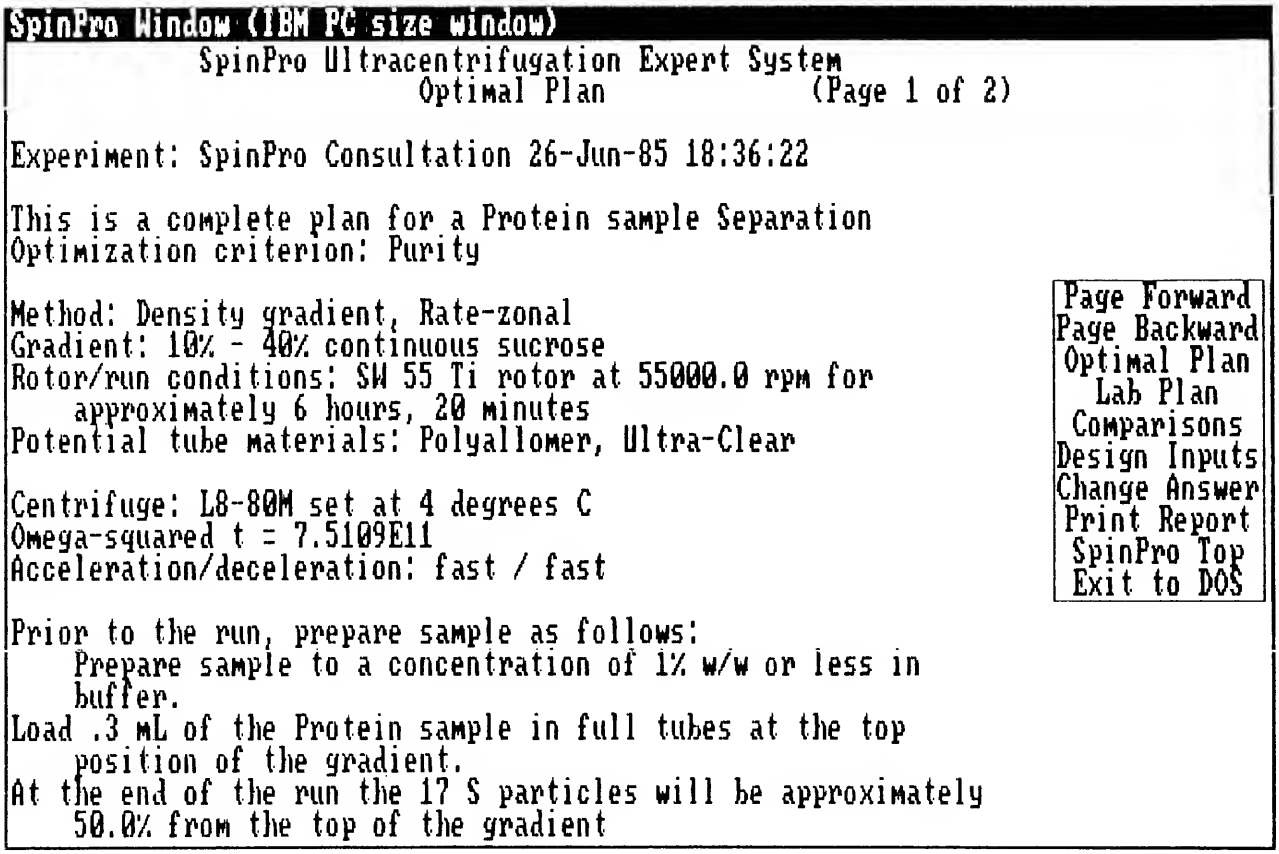
The Consultation function is the heart of SpinPro, as it is the actual expert system. During a consultation, when input is required from the user, the system will build a menu of values to choose from for multiple-choice questions, and will use a numeric keypad type menu for number-valued questions. All menu selections are made with the mouse. An Unknown option is available on all the

menus where such an answer makes sense. For example, it wouldn't really make sense to answer Unknown to a "Do you know ...?" type of question. Either you know or you don't. You can't not know if you know. At any question the user may ask the system Why it is asking this particular question, to which the system will provide a domain specific response, not simply a (probably meaningless) rule traceback. The user may also choose to Change an Answer to a previously asked question. SpinPro will then re-run the consultation remembering the answers given to previously asked questions and will use those answers rather than requiring the user to re-specify them.



SpinPro's Information hierarchy

Supporting the Consultation function of SpinPro are the Information, Calculations and Configuration functions. It sometimes happens that the procedure which is recommended in a consultation refers to some rotor technique, etc., that the scientist is not familiar with. Rather than simply referring the user to a rotor manual which may have been eaten by the creature from Joe's last recombinant DNA experiment, or leaving him on his own to find out about it, the Information function provides a graphical interface (a la GRAPHER) to a compendium of information about ultracentrifugation techniques, gradient materials, rotors, ultracentrifuges, etc., including bibliographic references for more detailed information. The Calculations function performs many of the common ultracentrifugation laboratory calculations, such as computing the Relative Centrifugal Force for a rotor at a specified speed. Finally, the Configuration function allows the scientist to inform SpinPro about the ultracentrifuges and rotors in the lab. This is used during a consultation to formulate the procedure constrained by the laboratory equipment.



Optimal plan report from a SpinPro Consultation

SpinPro should improve the performance of most ultracentrifugation laboratories. Our experience with SpinPro has shown that it can reduce run times and improve the quality of separations. The investigator can more easily design procedures precisely fitted to the particular requirements of the research. Supporting information and calculations are readily available. SpinPro addresses the problem of improving the preformance of the ultracentrifugation laboratory by working with the investigator to design efficient ultracentrifugation procedures.

Remember our Toll Free Numbers if you have software problems or questions.

(800)228-5325 outside California

(800)824-6449 inside California

GUIDON-WATCH
A graphic interface to a knowledge-based system

by Mark Richer
Stanford University
Computer Science Department
Knowledge Systems Laboratory
701 Welch Road, Bldg. C
Stanford, CA 94304
arpanet: RICHER@SUMEX-AIM

GUIDON-WATCH is a graphic interface to HERACLES (Heuristic Classification Shell), a tool for building knowledge-based consultation systems. HERACLES has evolved from several earlier systems including NEOMYCIN, EMYCIN, and MYCIN. More specifically, HERACLES is to NEOMYCIN as EMYCIN is to MYCIN. In contrast to EMYCIN, control knowledge in a HERACLES system (e.g., the diagnostic procdure used in NEOMYCIN) is represented separately and explicitly in a special language. GUIDON-WATCH provides an interface to a HERACLES knowledge base (e.g., NEOMYCIN knowledge base of Meningitis and other diseases) and allows a user to view both the collection of data and the forming, pursuing, and refining of hypotheses. The interface allows the user to watch the diagnostic strategy unfold. In essence, GUIDON-WATCH provides a user with an interactive and inspectable model of diagnostic reasoning, much in the same spirit as STEAMER provides a student with an interactive, inspectable model of a steam propulsion plant.

GUIDON-WATCH can ultimately benefit several different kinds of users including programmers, knowledge engineers, end-users (e.g., physicians), and students. Because we need tools ourselves, some of our effort has been directed towards making GUIDON-WATCH useful for our own system building and debugging. However, our current goal is to investigate strategies for teaching diagnosis to medical students. In keeping with this, GUIDON-WATCH is a first step towards an interface that will be useful to students learning diagnosis. Hopefully, GUIDON-WATCH will not only explain the behavior of NEOMYCIN during a consultation, but will help to reify the problem solving process (i.e., make it more concrete).

GUIDON-WATCH currently provides a user with over twenty windows that can be displayed to view a knowledge base and the current state of a consultation. Users can see a taxonomy of disorders, a causal-association network, a table of positive findings, hypotheses with evidence, and the current state or history of the diagnostic strategy. The interface was designed in a principled manner to help users cope with the complexity of the system and the numerous windows that are available. The mouse and menus are used in a consistent and simple manner. GUIDON-WATCH also manages the display (e.g., places windows) automatically for the user. In the fall of 1985, we will begin using GUIDON-WATCH and several other programs with medical students to collect data on its usefulness.

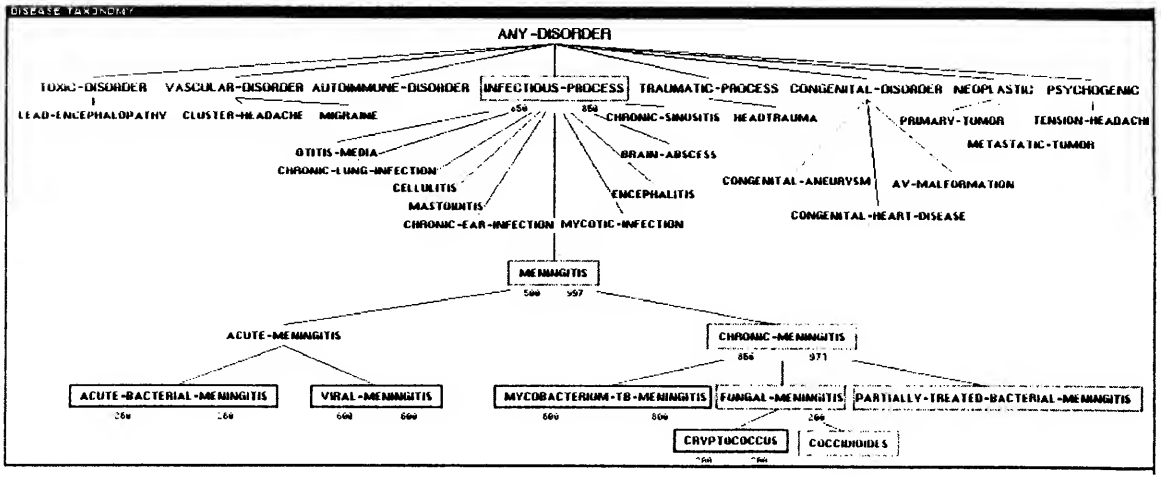
Two examples of the power of an interactive display are illustrated in the Taxonomy and Evidence windows (shown at the end of the article). These windows can be displayed at any time, but they are especially revealing during an actual consultation. During a consultation, dynamic information is collected such as disorders that are being considered, which findings are positive or negative, and which rules have succeeded or failed. In the examples below, dynamic information is displayed on top of static knowledge.

In the taxonomy window, a subtype hierarchy of disorders is displayed using the grapher package. During a consultation, when a hypothesis is added to the differential, its corresponding node in the Taxonomy window is flashed and then boxed. As evidence is gathered, the certainty factors for a hypothesis are printed underneath its node label. When a hypothesis is removed from the differential, a lighter box is drawn around it to leave a trace of the search path. This simple use of graphics makes it possible for a student to see how NEOMYCIN has searched the solution tree during a diagnosis. In particular, the student can clearly see how NEOMYCIN looks up in the tree before looking down and refining hypotheses.

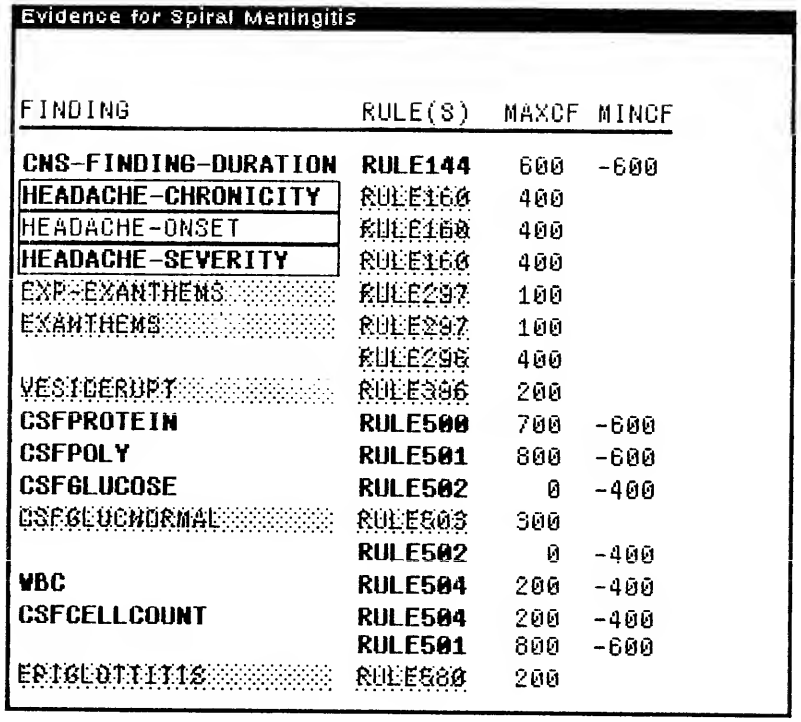
If a user buttons a node in the taxonomy tree, a pop-up menu is displayed that allows a user to select additional information including evidence, causal relations, etc. In the example below, the user had buttoned viral-meningitis, and then selected evidence from the pop-up menu. The evidence for a hypothesis is displayed in a table with findings printed in the first column and rules printed in the second column. Simply stated, the rule or rules listed in the second column use the finding in the first column to conclude about the chosen hypothesis. In the third and fourth columns, certainty factors may be printed that indicate the minimum and maximum certainty that the rule can conclude with. During a consultation, bold font printing in the evidence window is used to indicate a finding with a positive value or a rule that succeeded, while graying out a region is used to indicate a finding with a negative value or a rule that failed. This concise notation provides a viewer with a great deal of information quickly and in an easily understandable form. Future GUIDON programs will allow students to actively practice diagnostic skills. For example, GUIDON-MANAGE will allow students to choose a diagnostic operator (e.g., test hypothesis) and a focus (e.g., meningitis) while NEOMYCIN applies domain rules. This will provide students with the opportunity to think more explicity about strategic decisions during a diagnosis. A prototype of GUIDON-MANAGE will be completed this summer.

GUIDON-WATCH is described in more detail in a technical report (GUIDON-WATCH: A Graphic Interface for Browsing and Viewing a Knowledge-Based System, KSL-85-19) available from the author at the above address.

FIGURES



Taxonomy



Evidence for Spiral Meningitis

FINDING	RULE(S)	MAXCF	MINCF
CNS-FINDING-DURATION	RULE144	600	-600
HEADACHE-CHRONICITY	RULE160	400	
HEADACHE-ONSET	RULE160	400	
HEADACHE-SEVERITY	RULE160	400	
EXP-EXANTHEMS	RULE297	100	
EXANTHEMS	RULE297	100	
	RULE298	400	
VESICERUPT	RULE386	200	
CSFPROTEIN	RULE500	700	-600
CSFPOLY	RULE501	800	-600
CSFGLUCOSE	RULE502	0	-400
CSFGLUCNORMAL	RULE503	300	
	RULE502	0	-400
WBC	RULE504	200	-400
CSFCELLCOUNT	RULE504	200	-400
	RULE501	800	-600
EPIGLOTTITIS	RULE680	200	

Evidence

A KNOWLEDGE-BASED ENVIRONMENT FOR PROCESS PLANNING

by Chandra B. Mouleeswaran
and Hartmut G. Fischer

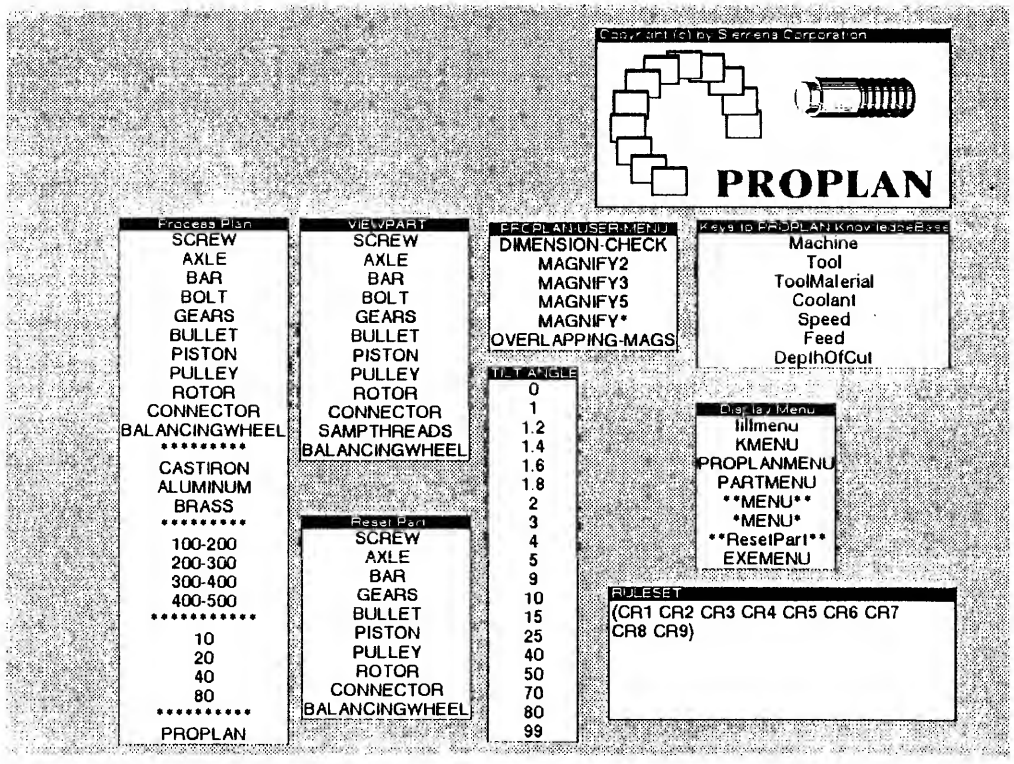
Siemens Corporate Research and Support, Inc.
Research and Technology Laboratories
105 College Road East
Princeton, New Jersey 08540

The system, implemented in Interlisp-D on Xerox 1108, integrates the design and the planning stages for manufacturing rotationally symmetric parts. To use the system, the user specifies a part in a simple language that describes the external and internal profiles of the part. These profile descriptors are composed of primitive segments LINE and ARC. A LINE is a straight edge on a part and is specified using the coordinates of the endpoints of the line. An ARC refers to curved surfaces on the part such as fillets and grooves, and is specified by the coordinates of the endpoints of the arc, the coordinates of the center of the arc, and the angle swept by the arc. These descriptors have been extended to include surface features, such as knurls, threads, and tolerance and finish requirements. A reference point is chosen for a part and the sequence of primitive segments is stored in list form for the profiles. Features of a part that cannot be described in profiles are accomodated as additional information.

The system estimates the raw material required, develops a sequence of machining operations to convert the raw material to finished parts, and provides detailed instructions for every operation, such as machining center, speed, feed, depth of cut, tool grade, tool material, number of rough and finish passes, coolant to be used etc. Inference trees are constructed with levels describing the gradient of the profile segment, the orientation of the profile segment and the shape of material between the workpiece profile and finished part profile. The frontier of these trees represents potential machining operations that can be applied to remove material from the raw block. For each machining process, the system generates a frame to hold the values of relevant parameters which are determined using production rules. If any conflict arises while planning is being carried out, the system presents alternatives to the user, and requests resolution. The system monitors the user's choices remembering them for future use. On selecting an operation, the system updates the workpiece profile and presents a graphical display of the state of the workpiece after applying the chosen operation. Each snapshot of the workpiece state is an active window and the system allows the user to inspect or modify details associated with that operation.

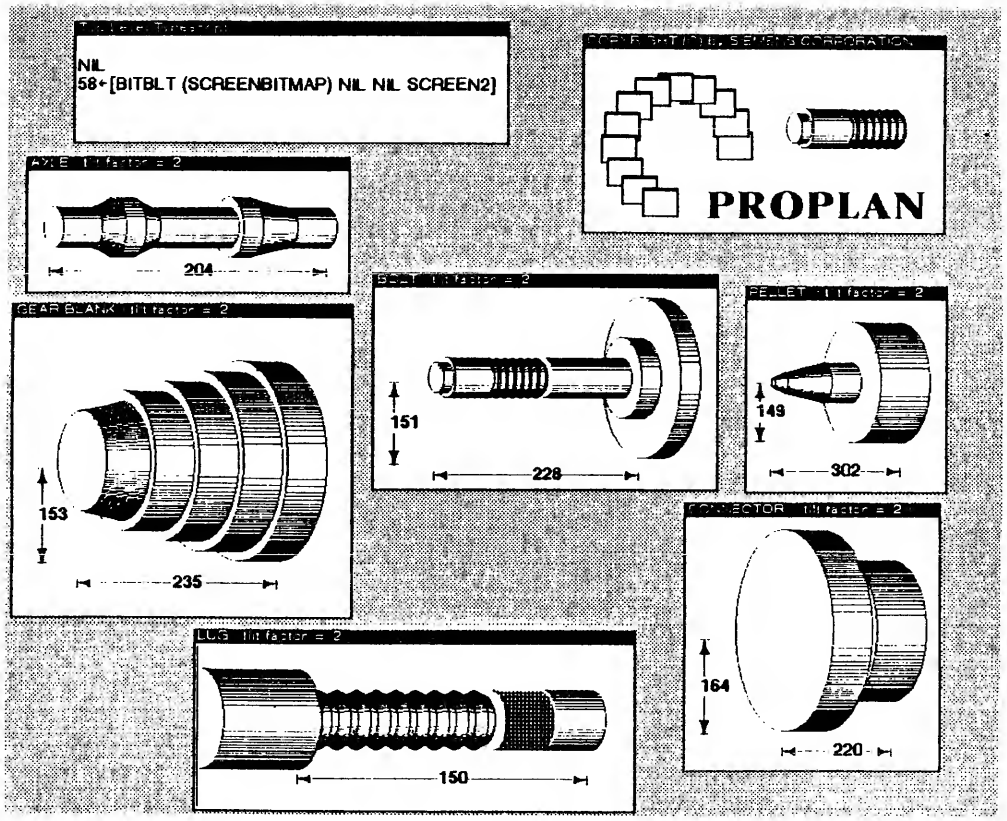
Our long term objective is to develop a methodology to combine a CAD system with manufacturing decision logic.

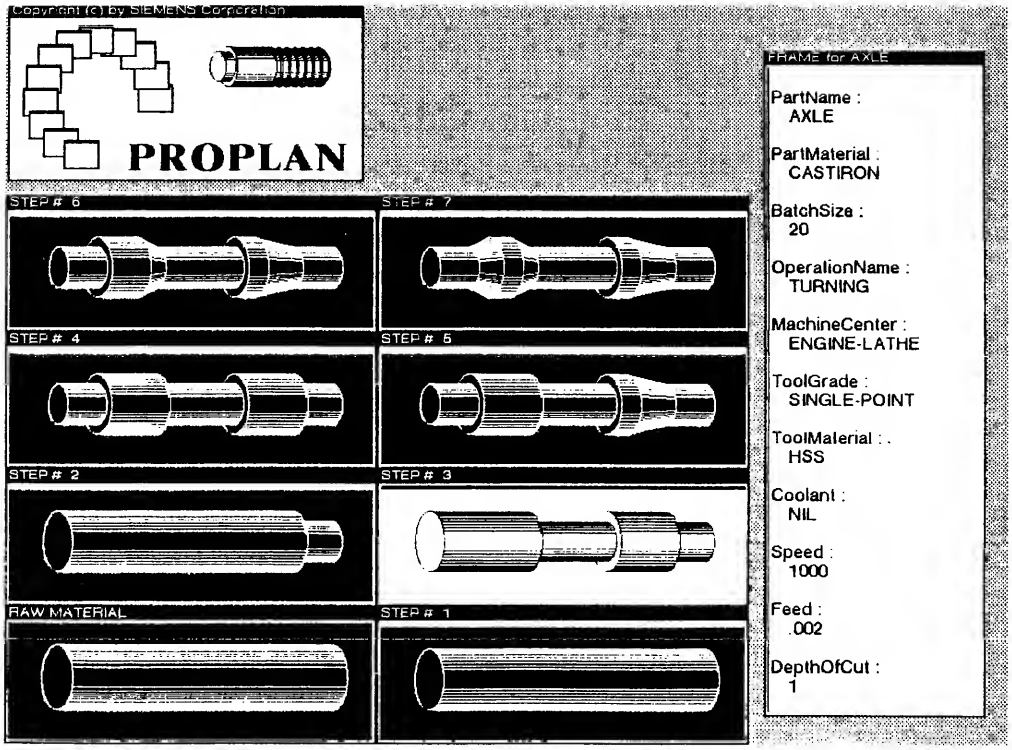
FIGURES



PROPLAN Menus

PROPLAN Sample Parts





PROPLAN Machining Sequence for a Part

Programmer's Corner :
Program and System Tips

Welcome to the Programmer's Corner, a column designed to benefit both the beginning and the more experienced Lisper. This column will also describe some of the recommended tactics concerning system use. In particular, this month's column has some very basic information. More advanced topics will be discussed in future issues of MASTERSCOPE.

- **Aborting:** Have you ever started up a function only to reconsider doing so a few seconds too late? In this case, you should stop cleanly.

  The way to do this in Interlisp is to type CONTROL-E, which unwinds the stack to the last ERRORSET. A CONTROL-B would work just as well (and would allow for continued computation if you so desired) but you would end up having to ↑ from the break. A CONTROL-D is often too destructive, especially if you're in several levels of DEdit because it "aborts the TTY process, and unwinds the stack to the top level," or nests several breaks into a composition you're degugging.

- **APROPOS:** It happens to everyone -- you're looking for some function or some variable that you know you've seen somewhere before and used before, and you remember the name of that item has xxxx (some string) in it but you don't remember the exact name. Simply type (APROPOS 'xxxx) and you will get a list of all VARS (and their values), PROPS and FUNCTIONS (and their argument lists) in the system that have that string in it. For example, (APROPOS 'DEFAULT) yields:

  DEFAULTPRINTINGHOST
  - Variable value: (Chronicle:)
  - Property list: (VALUE Guardian:)

  DEFAULTCURSOR - Variable value: ({BITMAP}#74,152732 0 . 15)

  DEFAULTFILETYPE - Variable value: TEXT

  DEFAULTEOFCLOSE - Variable value: NILL

  DEFAULTMAPFILE - Variable value: NIL

  PROC.DEFAULT.PRIORITY
  - Variable value: 2

  DEFAULTMENUHELDFN
  - Function arglist: (ITEM)

  and so forth....

Note that the string you supply can appear anywhere in the atom name, and all such occurences will be listed. One other note: if you give the 2nd parameter, ALLFLG, a value of T, then all system internal litatoms and functions will also be displayed.

- **? =** : Prints the argument names and their corresponding values for the current expression. This is especially useful when you don't want to wade through the Reference Manual to find them. For example, if you want to find the parameters for the function PUTPROP, enter the following:

  (PUTPROP ? = <cr> (* <cr> is the carriage return)

  (**PUTPROP** ATM PROP VAL) (* the function name and the parameter returned)

- **??:** If you've taken one of the Lisp Classes here, you will recall that a history list is kept of the last 100 commands that you've entered in the TOP LEVEL TYPESCRIPT WINDOW. If you wish to view this list, simply enter ?? at the prompt and the expressions you've entered as well as their results will be displayed. With this history list, one can UNDO or REDO a specific command that was executed on a specific line. For example, if you've DEdited a function, you can UNDO that and the function will be reset to its definition before the editing session.

  For more information about the history list, refer to the Interlisp Reference manual (Oct., 1983), Chapter 8.

- **Subitems in menus:** Since the Intermezzo release, you might have noticed that to the right of the HARDCOPY option in the background menu, there is a little arrow pointing to the right. If you slowly move your mouse in the HARDCOPY option just over and out of the menu, a submenu will pop-up. Submenus are easy to create. For example:

```
(LAMBDA NIL
  (ADDMENU (CREATE MENU
    ITEMS ← '((CALIFORNIA NIL NIL
                (SUBITEMS LOS.ANGELES BEVERLY.HILLS))
              ILLINOIS TEXAS)
    CENTERFLG ← T
    MENUCOLUMNS ← 1)))
```

The above hack creates the following menu and corresponding submenus:

Now, if you select California and move your cursor in that selection to the right, a submenu will appear:

Here is the general format:

```
[CREATE MENU
   ITEMS ←'(FOO (FOO2 NIL NIL (SUBITEMS sub2A sub2B))...)]
```

In place of the first NIL, you can put an expression that will be evaluated when the item is selected from the menu.

As for the second NIL, you can put a message there that will be printed in the Prompt Window when a button is held over the item.

Yes, subitems can have subitems themselves!

We encourage you to send any interesting programming tips that you would like to share with others in the user group to AINewsletter; please indicate that the sumission is for the Programmer's Corner.

Notes, Cautions and Helpful Hints

- **FX80STREAM:** The Lisp Library Package FX80Stream will allow you to print TEdit files to an Epson FX-80 Printer. (Model FX-100 works also). However, after loading the package, two things have to be set in order for the package to work properly. Evaluate the following once in every sysout :

  ```
  (RS232INIT 9600 8 NIL 1 'DTR)
  (SETQ RS232XON\XOFF? T)
  ```

  Please note that you will find the backslash "\" character on the key labeled 'FONT'.

  There are two sets of DIP switches on both the serial interface (upper) circuit board and the mother (lower) circuit board. These are labeled SW1 and SW2. The fourth switch in SW2 on the lower board must be set to ON to enable the printer to add <lf> to each <cr> sent. Note that this only applies when using FX80STREAM. Please read your Epson manual.

  The documentation for the FX80STREAM Lisp Library Package states: "To print a TEdit document, left-button "Hardcopy" on the TEdit window menu, slide off to the right, and pick "To a file...."

  It should read: "To print a TEdit document, left-button in the Tedit window's title bar and select "Hardcopy", slide off to the right of that selection and pick the "To a file" option....."

- **TTYIN:** When the HELP argument for TTYIN is T and the user types ?<cr> or HELP<cr>, then the help string in TTYIN is printed as expected but it returns NIL before one can respond. The workaround is:

```
(ADVISE 'TTYIN.FINISH 'AFTER'(AND(EQ !VALUE 'ABORT)
  (SETSTKARG 2 'TTYIN1)))
```

- **COPYFILES:** The Lisp Library Package Copyfiles was released with a change that was not documented properly: the "default" for copying files is to copy if the source version is DIFFERENT from the destination, not just NEWER. (This corresponds to a value of '#A rather than '>A).

- **RENAME:** This does not always use the current definition of the function which you are trying to rename. This bug is fixed by the file FILEPKGPATCH.DCOM on the floppy labled LIBRARY PACKAGES #1 . This file is not loaded by INTERMEZZOPATCHES.

- **NSCHAT:** NSCHAT may not fail gracefully when attempting to connect to dead hosts. The workaround is :

```
(ADVISE '(COURIER.OPEN IN NSCHAT.OPEN)
    'BEFORE
    '(SETQ NOERRORFLG T))
```

- **CMLARRAY:** The function AREF works fine when interpreted, but when compiled it gets changed into ASET. To fix this problem you should delete the BOGUSMACRO and DMACRO properties from the property list of AREF.

- **FTPSERVER:** Bugging INFO for the RTP process for FTPSERVER in the PSW still causes an 1108 to fall into 9305.

- **SKETCH:** If you are at a site without NS printers and are experiencing problems when loading SKETCH, then do the following:

  a. Before loading Sketch, for Modern font sizes: 8, 10, 12, 14, 18 and 24, do a
     (COPYFILES 'HELVETICA*-C0.DISPLAYFONT 'MODERN*-C0.DISPLAYFONT).
     COPYFILES is a Lisp Library package.

  b. Then, (RENAMEFILE 'MODERND24-C0.DISPLAYFONT 'MODERN24-C0.DISPLAYFONT).

- **MAXARGS:** The maximum number of arguments allowed in Interlisp-D is 80. This number is "hard-wired" into Lisp. The variable MAXARGS simply records this limitation as a globally accessible value.

- **Sanitization of the 1108 Hard Disk:** Xerox Corporation has received approval, from NSA, of a procedure for the sanitization of Xerox K95 10 MByte transportable Winchester type disk drive and M51 42 MByte transportable Winchester type disk drive used with various Secure Information Device products of the Xerox 8000 Network System. The procedure meets the specification requirements for overwriting disk media prior to release from SCI environments. Before using this procedure, customers should have their respective ADP System Security Representative (ADPSSR) submit the

procedure to their respective Contracting Officer's Security Representative (COSR) making the procedure an amendment to their approved ADP Security Plan.

The procedure is as follows:

1. Execute a full ALAG (boot #5 from the Diagnostic Disk).
2. Execute two (2) passes of DESTRUCTIVE SCAN TWO PASSES.
3. Reconfigure the disk using normal system installation procedures.
4. The two (2) DESTRUCTIVE SCAN procedures are to be executed by an individual who is fully cleared for the material contained on the disk, and witnessed by a second fully cleared individual. A letter describing the event should be forwarded to NSA ATTN T/SCSC for the record. (Additional notification may be required by specific COSR's).

The ALAG tool performs a full hardware verification including processor, memory, controllers, and disk drive.

The DESTRUCTIVE SCAN tool writes specific bit patterns across the entire disks surface. These bit patterns include: all ones, all zeros, alternate ones, alternate zeros, other arbitrary patterns. Each pattern written is subsequently verified and error messages are posted if problems are encountered.

- **1100 (Dolphins):** (STORAGE T) will crash an 1100 (not even control-shift-DEL accesses RAID) immediately prior to printing the allocated number of ArrayBlocks. Loading PRINTPATCH.DCOM will solve the problem.

  An 1100 with less than 1.5MB of memory cannot run Intermezzo. You will have to purchase additional memory or remain in Harmony or one of its predecessors.

  Isolated 1100 users shoule complete the Dolphin Users Survey that was sent out to customers in June. This will ensure they are scheduled for an upgrade. If you are an 1100 user and have not received the Dolphin Survey form, please contact 1100Support at our toll free number.

- **SELECTEQUAL:** SELECTQ uses EQ for comparisons. In some instances this comparison is too strict and an EQUAL comparison would be more appropriate. You can write your own version of SELECTEQUAL with the following:

```
[PUTPROPS SELECTEQUAL MACRO
  (X (bind (VAR ← (GENSYM))
        (SEL ← (pop X))
        while (CDR X)
        collect [BQUOTE
              ([AND ,@(for STR inside (CAAR X)
                    collect (BQUOTE (EQUAL ,VAR ,STR]
                                          ,@(CDR (pop X]
        finally (RETURN (BQUOTE
                        ([LAMBDA (, VAR)
                          (COND ,@ $$VAL (T ,@ X] ,SEL]
```

- **Programmer's Assistant:** to redefine ?? to only print expressions without the values. Use:

  (ADDTOVAR LISPXHISTORYMACROS
    (?? NIL (PRINTHISTORY LISPXHISTORY LISPXLINE NIL T T)))

  Then define ?V to print expressions and values with:

  (ADDTOVAR LISPXHISTORYMACROS
    (?V NIL (PRINTHISTORY LISPXHISTORY LISPXLINE NIL NIL T)))

- **PROMPTCHAR:** promptchar incorrectly inserts a <CR> at the end of the line forcing the user to make his entry at the next line rather than at the prompting line. The workaround is:

  (ADVISE '(FRESHLINE IN PROMPTCHAR) '(RETURN]
  (ADVISE 'ENTEREVALQT 'AFTER '(FRESHLINE T]

QUESTIONS and ANSWERS

Q. *Why does the Floppy driver get touched when I have not explicitly called it?*

A. Interlisp-D does not have hard-wired recognition of host names. One of the attributes ("methods") for a device driver is a host-name recognizer. When Interlisp-D needs to discover which device driver "knows about" a device, it asks each device in turn when is this likely to happen, e.g., at logout. This mechanism makes it very simple to add a new device driver.

Q. *In the past month user requests have arisen where they wanted to know the forgotten atom whose value cell prints to a particular window.*

A. The solution is to do

  (WHICHW) with the mouse cursor inside your window

  then

```
(MAPATOMS (FUNCTION
      (LAMBDA (A)
        (COND ((EQ (GETTOPVAL A) IT) (PRINT A].
```

Q. *What actually is a file?*

A. There are two distinct uses for the word "file".

Conventional usage is that a "file" is a collection of data stored in some physical (generally magnetic) media, or a physical-medium-file. Thus there are files on the local disk, files on the floppy, and files on a device called {CORE}.

Secondly, there is a "thing" that the file package deals with, which is a collection of Lisp program elements. These file-package-files contain functions, variables, records, and so forth, in a very specific format.

The process known as MAKEFILE takes a file-package-file and produces a physical medium file. Usually the "file-package-file" name is the root of the physical medium file. E.g., we talk about GRAPHER as a file-package-file, while {DSK}<Lispfiles>GRAPHER.DCOM;3 is the name of a physical-medium-file.

Q. *Is there a variable in Lisp that specifies the screen resolution in pixels/inch for the type of host machine? If not, do you know what it is for the 1100 series? I know it is about 90, but would like to obtain an accurate figure.*

A. The screen resolution is nominally 72 points per inch. In fact, most distances in Interlisp-D graphics are measured in terms of these units. The function (DSPSCALE NIL STREAM) will return the scale of a given STREAM in terms of screen points.

WE'VE BEEN EXPANDING OUR SALES FORCE AND WANT YOU TO KNOW THE CONTACT NUMBERS FOR YOUR LOCAL SALES RESPRESENTATIVE.

Arizona, Hawaii, Nevada, New Mexico, So. California, Utah:

Deborah Kelfer, Pasadena, CA.
(818)351-2351 OR:
(800)228-5325 outside California
(800)824-6449 inside California

Alaska, Idaho, Montana, No. California, Oregon, Washington, Wyoming

Steve Martino, Palo Alto, CA.
(415)494-5718

Joe Mazzarella, Tukwila, WA
(206)241-1200

Colorado, Iowa, Kansas, Minnesota, Missouri, Nebraska, No. Dakota, So. Dakota

Beth Derrough, Overland Park, KS.
(913)362-3800

Gerry Hatton, Golden, CO.
(303)278-8777

Alabama, Arkansas, Louisana, Mississippi, Oklahoma, Tennessee, Texas

Charles Jones, Richardson, TX
(214)907-0150

Federal Government/DC, Delaware, Maryland, Virginia, West Virginia

Wellington Pitts - Tom Marti - Jim Allen, Arlington, VA
(703)247-6952

New Jersey, New York, Pennsylvania

George Hunt, New York, NY
(212)916-2240

Connecticut, Maine, Massachusetts, New Hampshire, Rhode Island, Vermont

Arthur (A.J.) Matsis, Nashua, NH
(603)881-7028

Dave Dolbec, Hartford, CT
(203)275-6534

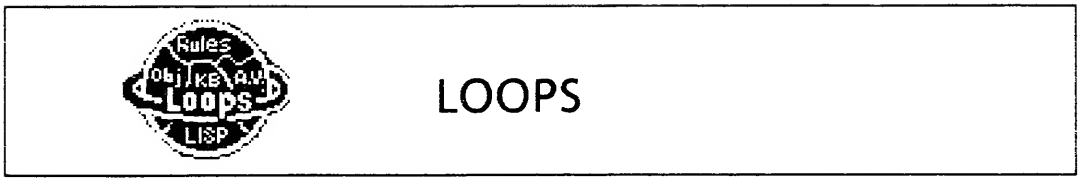
Illinois, Indiana, Kentucky, Michigan, Ohio, Wisconsin

Helen Godfrey - Mike Nelson - Scott Boucher, Arlington Heights, IL
(312)981-4252

(312)9811-4252

Florida, Georgia, No. Carolina, So. Carolina

Bruce Althoff, Atlanta, GA.
(404)934-1544

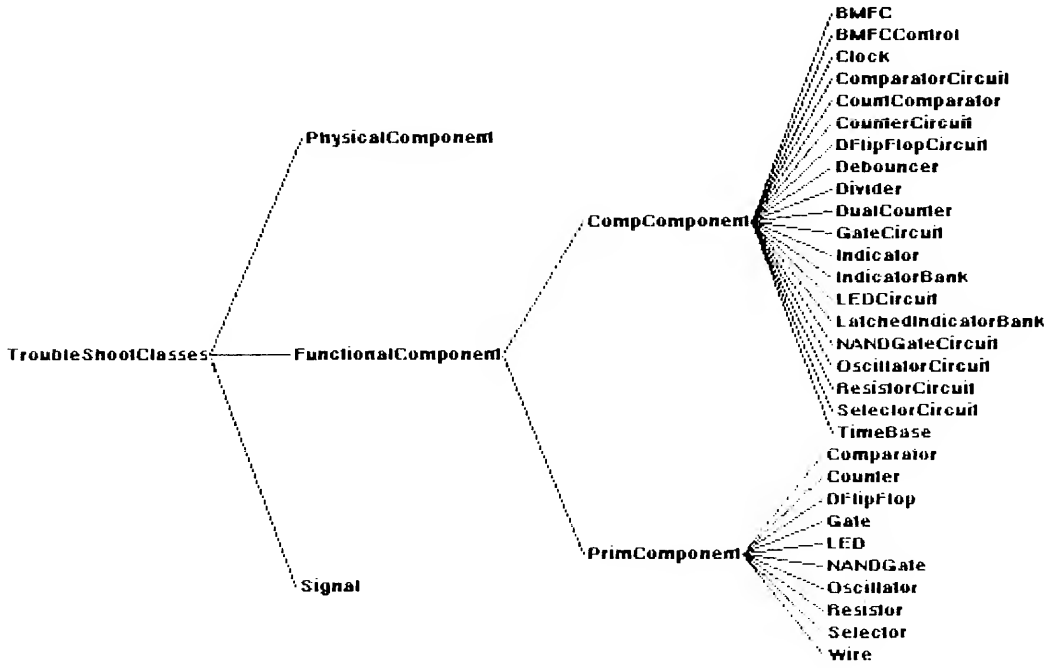


TSHOOT - A Recursive Expert Troubleshooting System

by Daniel Larner
Allen-Bradley Co.

As part of the software research program in AI within Allen-Bradley's Corporate Technology Development group, an experimental program aimed at the diagnosis and repair of systems was developed. The program is called TSHOOT, and it relies heavily on the capabilities provided by the LOOPS knowledge engineering environment.

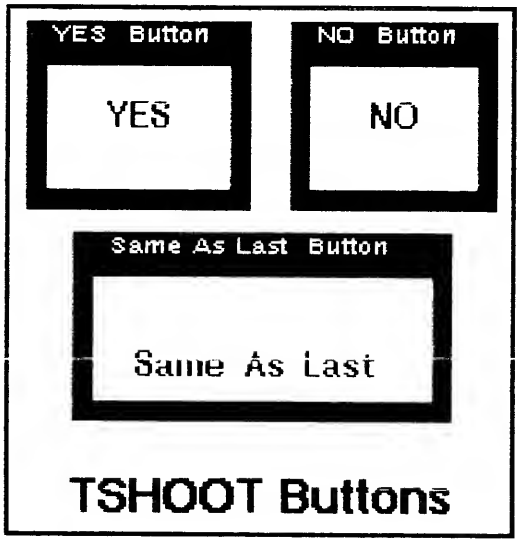
The TSHOOT approach towards troubleshooting is based on the concepts of hierarchical partitioning, connectivity / causality, function, and expert supplied heuristics. In a sense then, one could say that TSHOOT is a combination of 'Deep' level knowledge, and the 'Shallow' type of knowledge more commonly found in most current diagnosis systems.



TSHOOT Class Structure

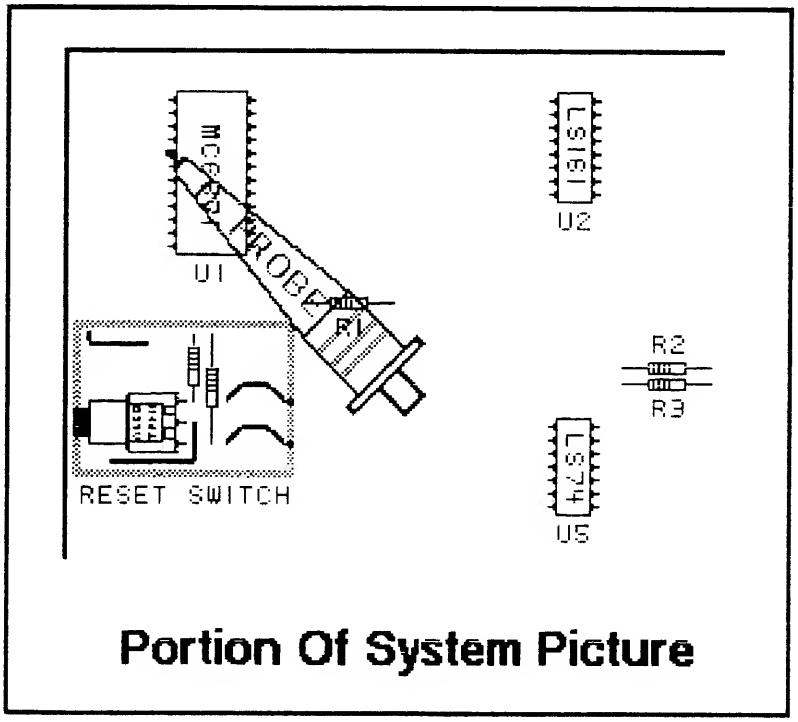
The object oriented programming methodology is utilized to capture the essential ideas of Signals, Physical Structure, and Composite and Primitive Components as depicted in the class structure. Class and Instance variables are used to hold information regarding connectivity and constructions, device specific heuristics and other attributes.

Methods are used to provide development support functions (such as saving, editing, etc.), and are also the mechanism which drives the system recursively through the component hierarchy. The session begins by sending an instance of the system a TroubleShoot message which invokes a three method strategy for traversing through components at a particular level in the device hierarchy. This strategy may in turn, recursively invoke itself at some deeper hierarchical level.

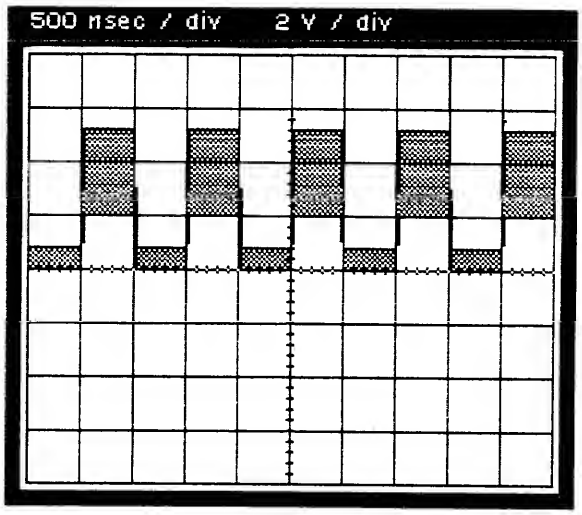


LOOPS Rulesets are used in many cases to simulate the behavior of the components. This ranges all the way from simulation of simple functions such as 'and' and 'or', all the way up to frequency comparators.

Active values provide useful support in the generation of component names and locations. They have also been used to drive simulations based on the availability of new signal information.



The user interface is augmented by the inclusion of a picture of the system under diagnosis. Drawings of test equipment probes, and LOOPS Gauges are placed on the system picture in order to precisely indicate test points, and show instrument readings. Also, a window that looks like the face of an oscilloscope is used to display relevant waveforms, and also to allow the user to draw waveforms present on the real test equipment. Additional interfaces include Yes/No button windows, and buttons to retrieve the last waveform drawn on the oscilloscope window.



Oscilloscope Window

The most difficult test case to date has been the successful troubleshooting of sixteen simultaneous faults (combinations of opens, shorts, and bad components) in an electronic circuit comprised of slightly more than a dozen MSI and SSI components.

The extreme power of the LOOPS integrated environment allowed the system to be developed to this level of expertise in about three man months.

While at IJCAI, plan to visit Xerox Artificial Intelligence Systems at our booth, and at our hospitality suite at the Beverly Hilton Hotel

EXPERT INSTRUCTIONAL SYSTEMS RESEARCH AT Learning Research and Development Center (LRDC)

Jeffrey Bonar, Alan Lesgold, and Stellan Ohlsson
University of Pittsburgh

The Intelligent Tutoring System (ITS) project at LRDC has focused on the issues of developing effective and practical ITS. LRDC is concerned with a wide range of educational issues, emphasizing detailed cognitive studies of learners, teachers, and curriculum. The ITS group has had two major goals:

1. Development of tutoring architectures that can be used by non-"Alexperts" to incorporate results from cognitive and educational studies of the domain to be tutored. We are looking toward the development of an ITS *"authoring language."* We intend to serve educators and cognitive science researchers who have little training in artificial intelligence or programming. Our goal is to allow such people to develop an effective and useful intelligent tutor by encoding their knowledge of a domain and how it is taught. In the research community at LRDC, for example, there are many people whose research focuses on detailed study of learners and subject matter.

   Although we do not know how to build intelligent tutor authoring languages for very sophisticated tutors or domains, we have begun development of a tutor architecture that could vastly simplify the development of intelligent tutors for constrained domains. Our architecture consists of a collection of *"bites,"* bite-sized chunks of the knowledge the tutor is to teach. These bites are interrelated to each other in relationships like *"more general form of," "prerequisite concept to,"* etc. Bites are also organized by a higher level layer of curriculum goals and pedagogical strategies. In general, bites give us the power to modularize a curriculum, packaging many of the traditional tutoring components -- e.g. problem generation, diagnosis, attribution of blame, etc. -- in general-purpose form.

   Bites are implemented as a hierarchy of LOOPS objects. The tutoring components are implemented as methods of those objects. We have used this architecture for tutors in the domains of multi-column subtraction, electricity, and economics (these tutors are discussed below).

2. Development of multiple instructional actions and intelligent "microworlds." For many people, the idea of computer based instruction (intelligent or otherwise) conjures up a dry presentation of facts, problems, and tutorial text, perhaps accompanied by graphics. Intelligence tutors add more sophisticated diagnostic and student modeling capability, but stick to a standard "problem, answer, diagnosis, tutoring, next problem" interaction cycle with the student.

   We are developing tutors that allow for multiple instructional actions. Our goals are tutors that, like a human teacher, can change not only the subject matter to be taught, but the specific form of the subject matter to teach. For example, our subtraction tutor can present a problem with detailed step-by-step coaching, a problem with minimal coaching, the same problem with a corresponding blocks display illustrating the place value notation, an interactive version of the blocks display which the student can use to manipulate the place value notation, and a series of drill exercises.

We are also developing tutors that allow a student to interact with a microworld of the domain being taught. These microworlds allow the student to directly interact with components of the domain under study. Our microworlds are designed to emphasize discovery, reflection on problem solving processes and knowledge structure, and problem solving with intermediate representations. A goal for our microworld based tutors is to simplify diagnosis of student errors. Because we are giving students a rich pathway to express their intentions to the tutor, the tutor can simplify the work needed to derive those intentions.

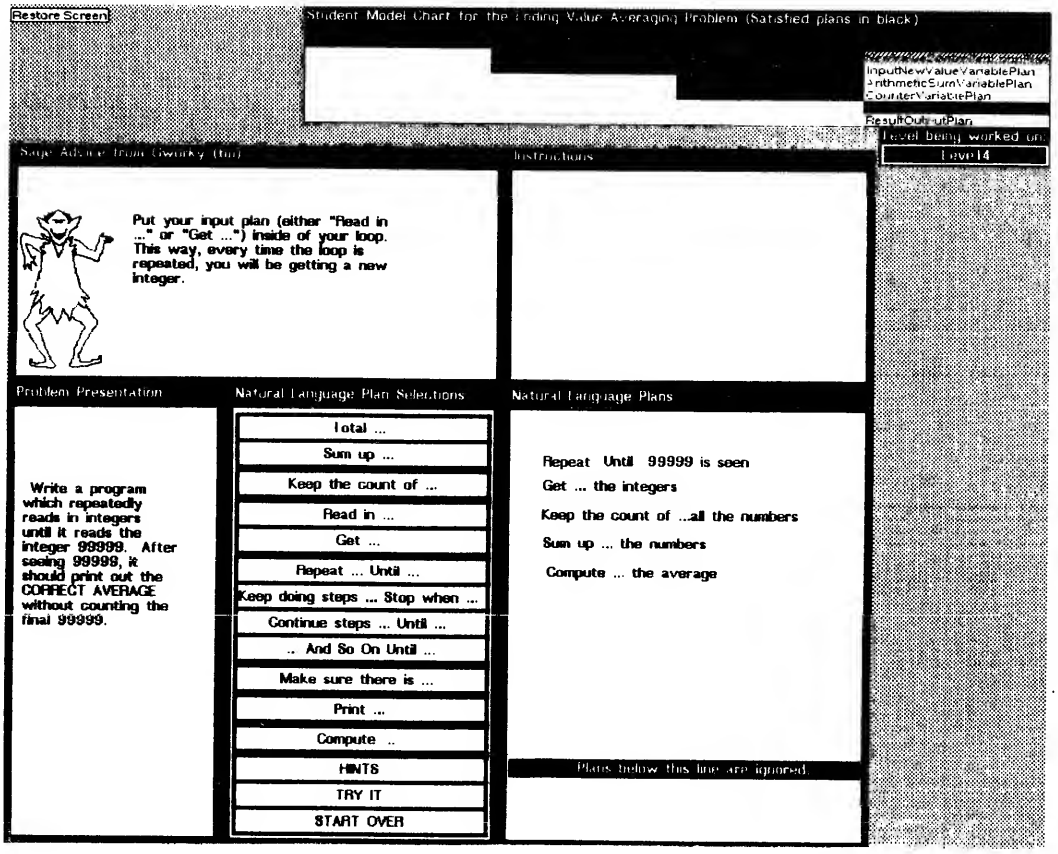
Our ITS are developed on XEROX 1108 workstations. We have made extensive use of the graphic display tools, user interaction tools, and LOOPS. These tools allow us to easily construct effective student displays. Just as important, they also us to construct powerful interface and design tools for our use. These tools allow us to watch what the tutor "knows", editing knowledge and tutorial components as the tutor is tested.

This work is supported by the Office of Naval Research, the Air Force Human Resources Laboratory, the National Science Foundation, and National Institute of Education.

Systems Under Development

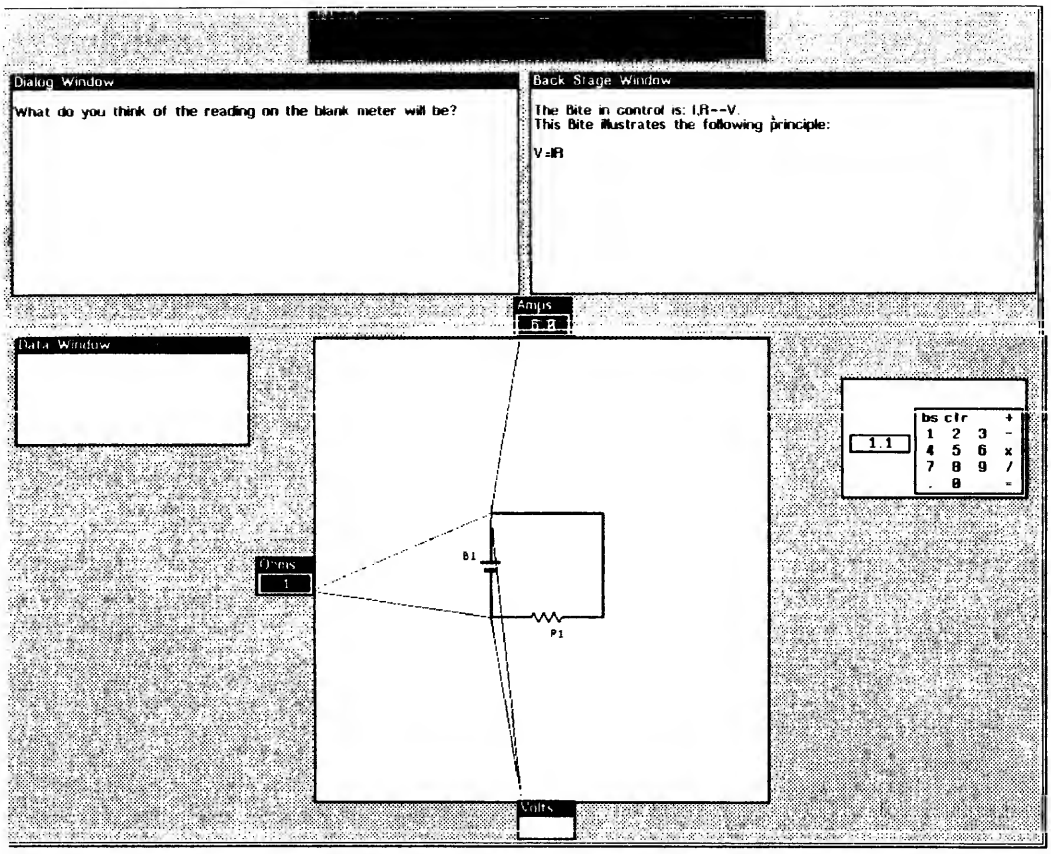
BRIDGE: A Computer Programming Tutor (with William Weil, John Corbett, and Bob Cunningham). A tutorial programming environment for beginning programmers. It coaches students in expressing a programming problem in a series of intermediate forms. Each intermediate form is progressively closer to an actual program. Programs are initially represented in natural language goals and schemata, then in programming language planning schemata, and finally in actual programming code. These intermediate forms are based on empirical studies of novice programmers solving programming problems.

BRIDGE



ELECTRICITY TUTOR: (with Cindy Cosic and Leslie Wheeler). This bite-sized tutor is designed to teach the basic laws of electricity. In this tutor we have emphasized a discovery approach that focuses on problem solving skills. The tutor begins with preliminary, simplified forms of each concept being taught. As these simplified forms of a concept are learned, the student must describe the subconcept involved. As a series of subconcepts are learned, the student must describe their relationship. In solving a multi-step problem, the student needs to describe the subconcepts used on the solution path.

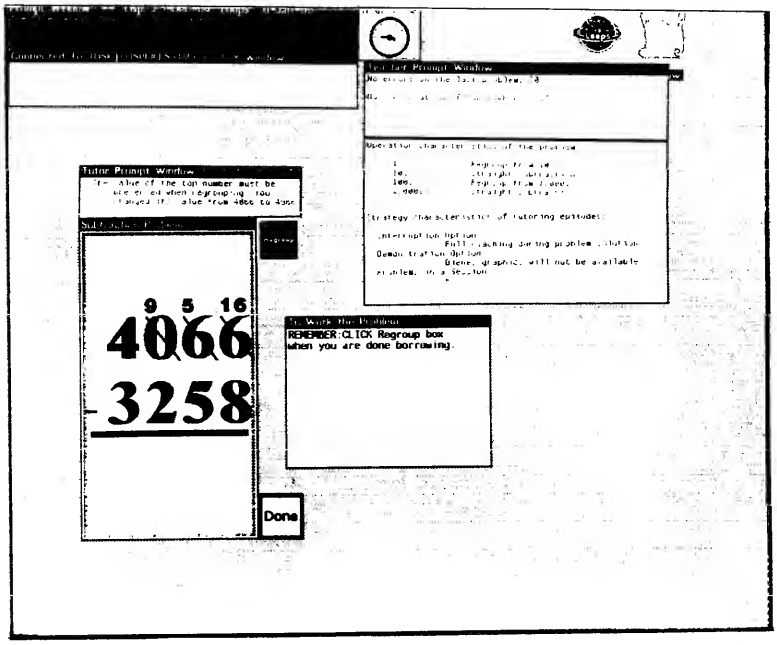
ELECTRICITY TUTOR



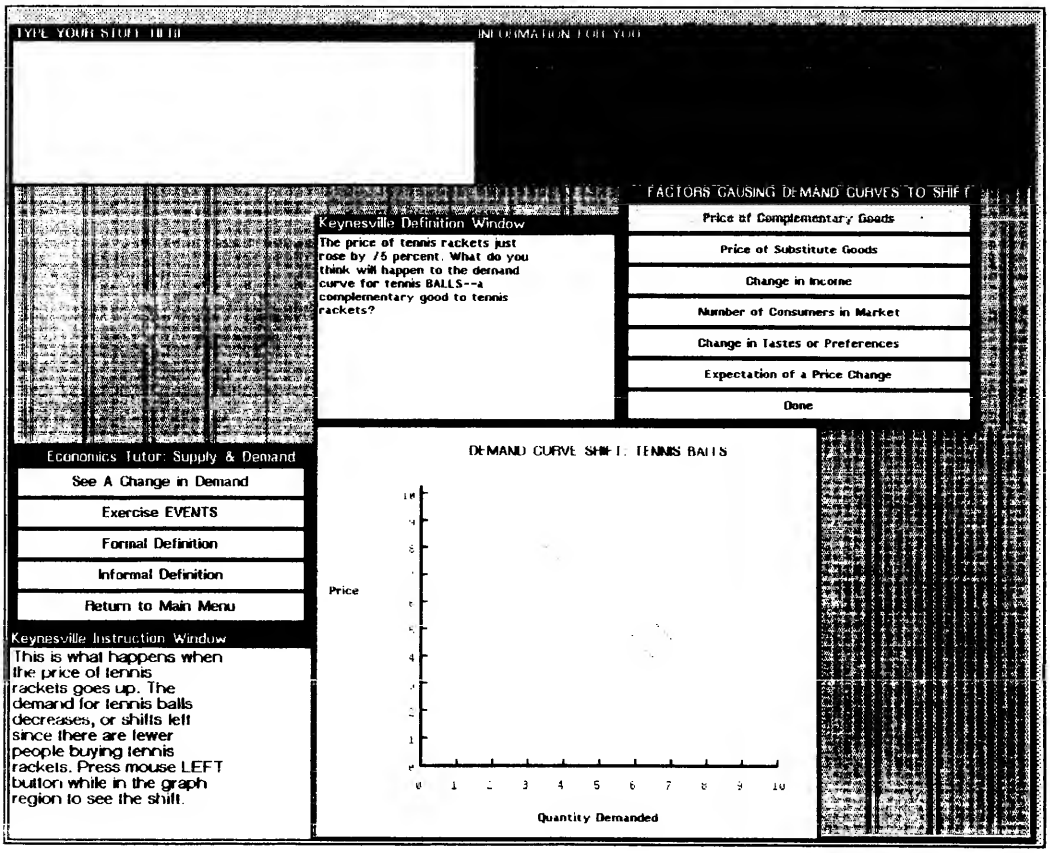
SUBITO: Multi-Column Subtraction Tutor (with Theresa Vitolo). Using a bite-sized architecture, we have built a tutor that combines several kinds of knowledge about teaching subtraction. In particular, one set of bites incorporates knowledge about the syntactic forms students find difficult (from Brown and Burton's BUGGY research). Another set of bites contain more semantic knowledge about numeric value and multi-column representation. The tutor also includes a Dienes block microworld.

SUBITO is a laboratory for several of the ideas mentioned above. In particular, we are implementing an interface that will allow non-programmers to specify the detailed instructional actions and curriculum of the tutor. Also, Stellan Ohlsson used this domain in developing a sophisticated diagnostic tool.

SUBITO



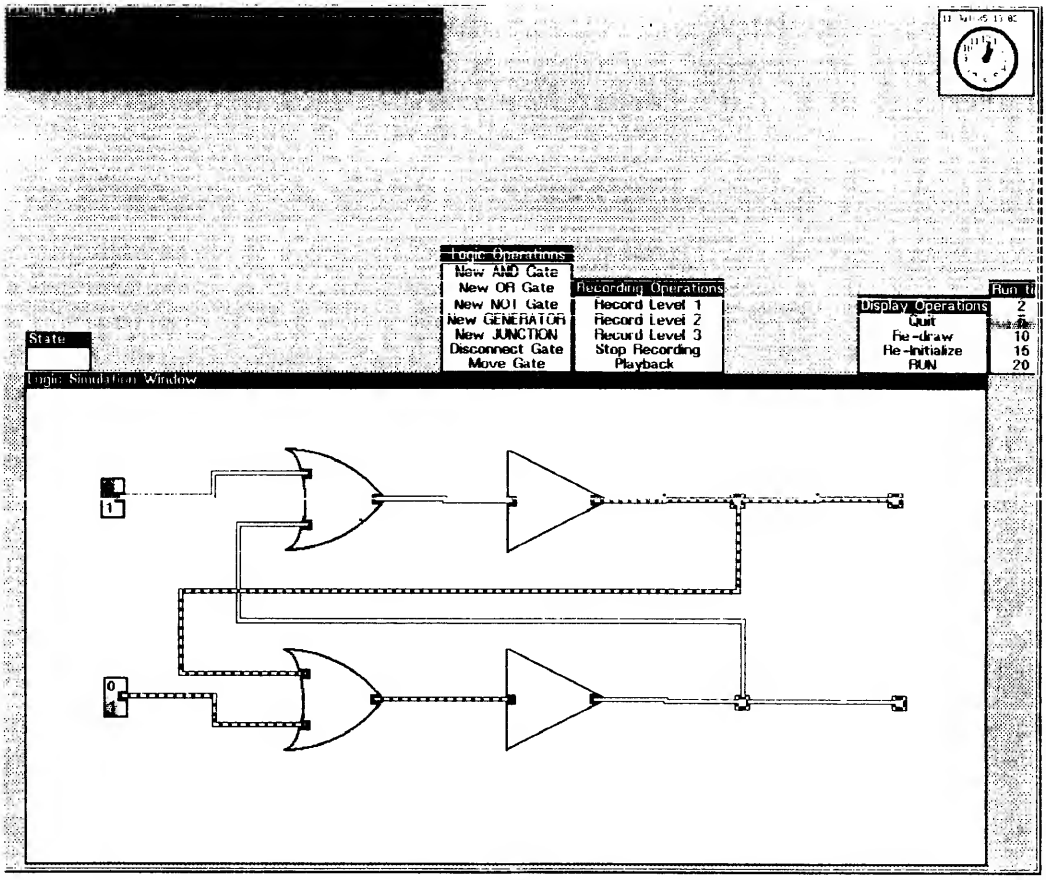
ECONOMICS TUTOR: (Valerie Shute and Paul Resnick). A bite-sized tutor where the bites represent various aspects of supply and demand theory. The tutor presents a microworld, the city of Keynesville, and generates its tutoring and problems in terms of various scenarios generated from the goods and actors within Keynesville. Students can pose their own scenarios to the system.



ECONOMICS TUTOR

Logic Microworld and FlowChips Tool (Marty Kent). An interactive digital logic simulation with an integral facility for detailed symbolic recording of the students actions. This microworld is a prototype for the FlowChips system - a set of LOOPS classes and methods for designing simulations of systems that can be modeled as flows between nodes.

Logic Microworld And FlowChips Tool



Optics Microworld (Peter Reimann). A microworld for discovery and simulation of basic optical laws. This microworld has featured the development of tools allowing students to record, analyze, and organize the results of experiments.

EUREKA: A Density Microworld (Leo Klopfer, Matthew Lewis, Ted Reese). A microworld for discovery, simulation, and experimentation with laws of density. The tutor is based on a series of problems posed to the student. In each problem the student makes a prediction, explanation, and then performs a simulated experiment to test their prediction. The tutor proposes problems designed to expose and correct common misconceptions and naive models of density.

Announcements

Submissions to Newsletter

Please send submissions for future issues and requests to be added to the mailing list to:

Via network:

AiNewsletter ↑ .pasa@Xerox

Via US Mail:

AINewsletter
ms 1232
Xerox Artificial Intelligence Systems
250 North Halstead Street
Pasadena, California 91109

(Notice that the name of our organization has changed.)

Software Support

If you have 1108s and have not received the Intermezzo release please contact Software Support.

You will soon be receiving DISKDLIONPATCH.DCOM in the mail. This will correct the following problem experienced by some 1108's with 1.5 mb memory: After fetching a fresh sysout and booting into Lisp from Tajo, Lisp hangs while loading TEDIT. The MP panel still reads 1108 and the diagnostic floppy finds nothing wrong. If one then attempts to do a 0 or 1 boot, the machine hangs with an MP code of 149 (disk not ready). The problem is that a function definition used in 1108 disk access is being swapped out.

The VMS modem software for version 4.0 is now available.

For Xerox software support, messages can be sent to 1100Support.pasa@Xerox. Our toll free numbers are:

(800) 228-5325 -- US, including Hawaii and Alaska
(800) 824-6449 -- within California

MAX-AI

On Tuesday, June 25, MAX-AI (Mid-Atlantic Xerox AI) Users Group convened in Washington, D.C. This is our first Users' Group to be started. The 50 or so participants agreed to bi-monthly meetings, the first of which will be held on July 30, 1985 at the Planning Research Corporation (PRC) in Mclean, Virginia. If you are interested in participating, please call at (703)274-5540 or write:

Dr. Joseph Psotka
US Army Research Institute
ATTN: PERI-ASO
5001 Eisenhower Avenue
Alexandria, VA. 22333

Sources

Selected sources are available for $495. Contact your Marketing Representative for a complete listing and details for obtaining the sources.

Xerox Third Party Software Catalogue

We are now preparing a listing of software available from third party vendors. If you have anything you would like to offer in this catalogue, please contact AINewsletter at the above address or call us.

Interlisp and Loops Training Classes

Xerox Artificial Intelligence Systems offers Interlisp-D and Loops classes. Each class is one week in duration, and includes both lecture and lab. Below is the training schedule for the remainder of 1985. Classes will be held in Pasadena, California, Leesburg, Virginia, and Chicago, Illinois.

DATE (week of..)	CLASS	LOCATION
July 29	Introduction to Interlisp-D	Pasadena
August 5	Intermediate Interlisp-D Programming	Pasadena
August 12	Programming in Loops	Pasadena
September 16	Introduction to Interlisp-D	Leesburg
September 23	Intermediate Interlisp-D Programming	Leesburg
September 30	Programming in Loops	Leesburg
October 14	Intermediate Interlisp-D Programming	Pasadena
October 21	Advanced Interlisp-D Programming	Pasadena
October 28	Introduction to Interlisp-D	Chicago
November 4	Intermediate Interlisp-D Programming	Chicago
November 11	Intermediate Interlisp-D Programming	Leesburg
November 18	Advanced Interlisp-D Programming	Leesburg
December 2	Introduction to Interlisp-D	Pasadena
December 9	Intermediate Interlisp-D Programming	Pasadena
December 16	Advanced Interlisp-D Programming	Pasadena

Contact the Training Coordinator at extension 2676 at the above toll free numbers for more information.

Xerox AI International Users' Group

We are having our first International Users' Group meeting during IJCAI in Los Angeles. This is your chance to meet many of the other users from around the world, ask us questions, give us feedback about how we are doing, and hear us discuss where we are going. Among others, the participants will include Gary Moskovitz, General Manager of Xerox AIS, Beau Sheil, Director of Product Development, and Dennis Dunn, Director of Marketing. If you haven't signed up for the Users' Group meeting send the following registration form to:

AINEWSLETTER
MS 1232
Xerox Artificial Intelligence Systems
250 North Halstead Street
Pasadena, California 91109

Name: _____

Organization: _____

Address: _____

City: _____

Phone: _____

_____ Yes, I will attend the Xerox hosted meeting/dinner on August 21

_____ No, I will not be able to attend.

I would like to be involved with the:

_____ Editorial Board for the Users' Group Newsletter; Masterscope

_____ User' Group Organization

- _____ As an Officer
- _____ On the International Organizing Committee
- _____ On the Local (Chapter) Organizing Committee

Space is limited, so sign up early.

WE LOOK FORWARD TO SEEING YOU AT IJCAI

AINEWSLETTER
M.S. 1232
Xerox Artificial Intelligence Systems
250 North Halstead Street
Pasadena, California 91109